APPLICATION FOR
UNITED STATES LETTERS PATENT

Title: **Method and Apparatus for Ascertaining the Status of Multiple Devices Simultaneously Over a Data Bus**

Inventor(s): **Robert B. Smith**
Edward A. Cross

Method and Apparatus for Ascertaining the Status of Multiple Devices Simultaneously Over a Data Bus

Cross Reference to Related Applications

This application is related to a commonly-owned and concurrently-filed patent application entitled "Method and Apparatus for Addressing Multiple Devices Simultaneously Over a Data Bus," which is hereby incorporated by reference in its entirety.

BACKGROUND

Field of the Invention

The present invention relates to data communications in computer systems and, more particularly, to improved data communications among computing devices using a data bus.

Related Art

A conventional computer system typically includes a central processing unit (CPU), main memory, and a number of devices that are in communication with each other and the CPU over a data bus, sometimes referred to as an Input/Output (I/O) bus. The CPU, for example, transmits commands and data to the devices (such as hard disk drives, printers, and displays) over the data bus, and vice versa. A variety of conventional data buses exist, such as the Small Computer System Interface (SCSI) bus, the Industry Standard Architecture (ISA) bus, the Peripheral Component Interface (PCI) bus, and the Inter-IC (I^2C) bus.

Devices typically communicate with each other over a data bus using messages that are defined according to a predetermined protocol. Furthermore, each of the devices is

typically assigned a unique bus address. Each message that is transmitted over the data bus addresses the device that is the target of the message using the target device's unique bus address. Each of the devices on the bus typically determines whether a particular message is addressed to it by examining the address specified by the message and comparing it to the device's unique bus address.

Data buses may include any number of data lines, each of which is responsible for transmitting one bit of information at a time. The number of data lines in a bus is typically referred to as the "width" of the bus. Typical bus widths range from 1 to 64 bits. The wider the bus, the higher the potential throughput for a given bandwidth or clock rate. Data buses with one data line are generally referred to as "serial" buses, while data buses with two or more data lines are generally referred to as "parallel" buses.

Parallel buses are typically used when high-speed data transfer is required. Wide data buses, however, are relatively expensive and are often difficult to implement over long transmission distances.

Serial buses are relatively inexpensive and are ideal for implementing long-distance data transmissions. In a serial bus architecture, each bit of a data byte or word (referred to herein as a "datum") is sent sequentially over the serial bus's single data line until transmission of the datum is complete. The protocol associated with a serial bus typically specifies how data transmitted over the bus are to be delimited and how the start and end of each data transfer is to be identified.

The sequential nature of data transfer over a data bus imposes limitations on the speed with which information may be transmitted over the bus. In particular, the speed with which information may be transmitted over a serial bus is limited by the fact that there is only a single data line over which

individual bits are transmitted sequentially. In addition, it is typically only possible to address one device on a serial bus at a time. Addressing multiple devices on a serial bus typically requires addressing each of the multiple devices in sequence.

It is sometimes desirable to ascertain the status of a plurality of devices coupled to a data bus so that, for example, a particular device needing attention may be identified. This can be difficult to do quickly, particularly if many devices coupled to the data bus are candidates for attention, because it typically is necessary to sequentially communicate with each candidate device to ascertain the status of each device so that the device needing attention may be identified. Such sequential communication can be time-consuming, particularly when a serial bus is being used.

What is needed, therefore, are techniques for more efficiently ascertaining the status of multiple devices coupled to a data bus.

SUMMARY

In one aspect of the present invention, techniques are provided for simultaneously ascertaining the status of a plurality of devices coupled to a data bus. For example, in one embodiment a method is provided that may be performed by a master device to ascertain the status of a plurality of slave devices on the data bus simultaneously. The method includes steps of: (A) transmitting a status request message over the data bus to the plurality of slave devices; and (B) receiving over the data bus a status indicator message including a plurality of status indicators indicating statuses of the plurality of slave devices. The master device may ascertain the status of at least some of the plurality of slave devices by examining the status indicators.

The data bus may, for example, be a serial data bus such as an I^2C bus. The status request message and/or the status indicator message may be a message defined according to a protocol associated with the data bus (such as the I^2C protocol). Each of the status indicators may, for example, be a single bit, such as an IRQ status bit.

The plurality of slave devices may receive the status request message over the data bus from the master device and collectively transmit the status indicator message to the master device over the data bus. Each of the plurality of slave devices may provide at least one of the plurality of status indicators to the status indicator message. For example, in one embodiment, each of the status indicators is a single status bit provided by a particular one of the slave devices. The status bits are combined into one or more status indicator bytes that are transmitted to the master device.

Other features and advantages of various aspects and embodiments of the present invention will become apparent from the following description and from the claims.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a schematic block diagram of a computing system including a data bus and a plurality of computing devices according to one embodiment of the present invention.

FIG. 2A is a flowchart of a method that is used by a master device to transmit a message over a data bus to one or more slave devices according to one embodiment of the present invention.

FIG. 2B is a flowchart of a method that is used by a slave device to receive a message transmitted over a data bus to one or more slave devices according to one embodiment of the present invention.

FIG. 3A is a timing diagram of signals transmitted over an I^2C bus during transmission of a message over the bus.

FIG. 3B is a timing diagram of signals transmitted over an I^2C bus during transmission of a message over the bus according to one embodiment of the present invention.

FIG. 4A is a flowchart of a method that is used by a master device to ascertain the status of a plurality of slave devices according to one embodiment of the present invention.

FIG. 4B is a flow chart of a method that is used by a plurality of slave devices to indicate their status to a master device according to one embodiment of the present invention.

FIG. 4C is a flowchart of a method that is used by each of a plurality of slave devices to indicate its status to a master device according to one embodiment of the present invention.

FIG. 5 is a functional block diagram illustrating the transmission of a status indicator message from a plurality of slave devices to a master device over a data bus.

DETAILED DESCRIPTION

In one aspect of the present invention, techniques are provided for ascertaining the status of multiple devices on a data bus (such as a serial data bus or a parallel data bus) simultaneously. In particular, techniques that may be used by a master device to transmit a status request message over a data bus. The status request message may be simultaneously received by a plurality of slave devices on the data bus. The plurality of slave devices, in combination, may respond to the status request message by transmitting over the data bus a status indicator message that includes a plurality of status indicators indicating statuses of the plurality of slave devices. The master device may receive the status indicator message and thereby ascertain the status of the plurality of slave devices. The master device may also examine the status

indicator message and thereby determine which of the plurality of slave devices requires attention.

Examples of techniques that may be used by the master device to transmit the status request message to the plurality of slave devices simultaneously are described in the above-referenced patent application entitled "Method and Apparatus for Addressing Multiple Devices Simultaneously Over a Data Bus." In brief, each device on the bus may have both a primary address and a secondary address. A plurality of devices on the bus may share a common primary address. A master device on the bus may simultaneously address the plurality of devices by addressing them using the common primary address according to the bus protocol. The master device may address a subset of the plurality of devices using a secondary address associated with the subset.

More generally, the master device may transmit a message (such as the status request message described above) to the device(s) corresponding to a particular primary-secondary address pair by first addressing the device(s) using the primary-secondary address pair (by transmitting the address pair over the bus) and then transmitting the content of the message over the bus. Devices on the bus are configured to receive and process messages that are addressed to them using primary-secondary address pairs.

The plurality of slave devices may transmit the status indicator message by, for example, each transmitting a single status indicator (such as a single status indicator bit) over the data bus. These status indicators may be combined (e.g., into a single byte) to form the content of the status indicator message that is transmitted to the master device.

The techniques described generally above and described in more detail below may be used to efficiently ascertain the status of multiple devices on a data bus. In particular, the ability to transmit a status request message to a plurality of

slave devices simultaneously and the ability of the slave devices to transmit a status indicator message that simultaneously indicates the status of the plurality of slave devices enables the master device to ascertain the status of the plurality of slave devices more quickly than techniques that require the master device to communicate sequentially with each slave device.

Referring to FIG. 1, a schematic block diagram is shown of a computer system 100 including a data bus 102 and a plurality of computing devices 104a-f according to one embodiment of the present invention. Although the data bus 102 may be any data bus, in one embodiment the data bus 102 is an Inter-IC (I^2C) bus. The architecture and operation of the I^2C bus is defined in a document entitled "The I^2C-Bus Specification," available from Philips Semiconductors. Version 2.1 of "The I^2C-Bus Specification," dated January 2000, is incorporated by reference herein in its entirety. Although the I^2C bus is a serial data bus, the data bus 102 may be either a serial bus or a parallel bus.

The devices 104a-f that are coupled to the data bus 102 are capable of communicating with each other over the data bus 102 using a physical architecture and communications protocol associated with the bus 102. The I^2C bus specification referenced above, for example, defines such a physical architecture and communications protocol. The devices 104a-f include data bus interfaces 106a-f, respectively, for communicating with each other over the data bus 102 in accordance with the bus's architecture and protocol. The data bus interfaces 106a-f may, for example, be standard I^2C hardware interfaces. One such interface is provided in the model PIC16C72a microprocessor available from Microchip Technology Inc. of Chandler, Arizona.

As described above, each device that is coupled to a bus is typically provided with a unique address on the bus. The

first datum in a message transmitted over a bus is typically an address. Each device on the bus receives and examines the address to determine whether it is the device's unique bus address. If a device determines that the address transmitted over the bus is the device's unique bus address, the device receives and processes the remainder of the message. Otherwise, the device ignores the remainder of the message.

As shown in FIG. 1, conventional devices 104e and 104f are provided with unique addresses 124a and 124b, respectively. As a result, conventional device 104e may be addressed by other devices over the bus 102 using address 124a, and conventional device 104f may be addressed using address 124b. For example, when a message is transmitted over the bus 102, the data bus interface 106e determines whether the address specified in the message matches the address 124a, and the data bus interface 106f determines whether the address specified in the message matches the address 124b. Further details of the operation of the data bus interfaces 106e-f are well-known to those of ordinary skill in the art and will therefore not be described herein.

As described above, in a serial bus architecture such as the I^2C bus architecture, it is typically only possible to address one device on the bus at a time. For example, the conventional device 104e may be addressed using the device's address 124a. To address multiple devices on a serial bus, it is typically necessary to address each of the devices individually in some sequence.

In one aspect of the present invention, techniques are provided for ascertaining the status of multiple devices on a data bus simultaneously. Simultaneously ascertaining the status of multiple devices may involve transmitting messages to multiple devices simultaneously. Examples of techniques that may be used to perform such simultaneous message transmission are described in more detail below with respect

to FIGS. 2A-3B and in the above-referenced patent application entitled "Method and Apparatus for Addressing Multiple Devices Simultaneously Over a Data Bus."

Referring to FIGS. 4A-4B, techniques will now be described for ascertaining the status of multiple devices simultaneously over the bus 102. Referring to FIG. 4A, a flowchart is shown of a method 400 that may be used by one of the devices 104a-d (referred to herein as the "master device") to ascertain the status of a plurality of other ones of the devices 104a-d (referred to herein as "slave devices") simultaneously. Referring to FIG. 4B, a flowchart is shown of a method 450 that may be performed collectively by a plurality of the slave devices 104a-d to transmit over the bus 102 a status indicator message that indicates the statuses of the slave devices. It should be appreciated that any one of the devices 104a-d may become a master device at a particular time by initiating the transmission of a message over the bus 102, in which case the remaining devices are considered slave devices for purposes of receipt of the message.

Referring again to FIG. 4A, the master device transmits a status request message over the bus 102 to a plurality of slave devices (step 402). In one embodiment, the status request message is a single message defined according to a protocol associated with the data bus 102. The plurality of slave devices may be all of the other devices on the bus 102 or a subset of the other devices on the bus 102. Examples of techniques for sending the status request message to the plurality of slave devices simultaneously are described in more detail below with respect to FIGS. 2A-3B.

Referring to FIG. 4B, each of the addressed slave devices receives the status request message transmitted by the master device (step 452). In response to receiving the status request message, the slave devices transmit to the master device over the bus 102 a status indicator message indicating

the statuses of the slave devices (step 454). In one embodiment, the status indicator message is a single message defined according to a protocol associated with the data bus 102. The status indicator message may, for example, include a plurality of status indicators, each of which may indicate a status of a particular one of the slave devices. In response to receiving the status request message, for example, each slave device may transmit over the bus a status indicator indicating the slave device's status. These status indicators may be combined into a single status indicator message that is transmitted in step 454, as described in more detail below with respect to FIG. 5.

Referring to FIG. 4C, a flowchart of a method 460 is shown that may be executed by each of the slave devices to implement the method 450 illustrated in FIG. 4B. Each slave device receives the status request message from the master device (step 462). Each slave device transmits a status indicator to the master device indicating the slave device's status (step 464). The combination of status indicators transmitted by the slave devices forms the content of the status indicator message that is transmitted to the master device in step 454 of FIG. 4B.

The content of the status indicator message may, for example, be a single byte or word, and each status indicator in the status indicator message may be a bit at a particular bit position in the byte or word. The status indicator message may also include multiple bytes or words that include status indicators. Assume for purposes of example that the status indicator message is implemented using a single 8-bit byte.

For example, referring to FIG. 5, a functional block diagram is shown of an embodiment in which the status indicators are implemented as bits to form a single byte. Slave devices 506a-g collectively transmit a status indicator

message 504 (step 454 in FIG. 4B) over the data bus 102 to a master device 502. The content of the status indicator message is a single byte. More specifically, each of the slave devices 506a-g transmits a single status indicator bit in the status indicator message 504 (FIG. 4C, step 464). The value of the status indicator bit transmitted by each slave device indicates the status of the slave device and is positioned within the status indicator message 504 at a bit position corresponding to the secondary address of the slave device. Note that slave devices 506a-g may include controllers and data bus interfaces and otherwise operate in the same manner as the devices 104a-d shown in FIG. 1.

For example, slave device 506a has a secondary address of 7 (binary 111), and therefore transmits its status indicator bit (having a value of one in this example) at bit position 7 in the status indicator message. In a serial bus architecture, the status indicator message 504 may be transmitted over the bus as a sequence of bits. Note that in this embodiment, bit zero of the status indicator message 504 is unused because secondary address 0 is reserved to address all of the devices sharing a particular primary address.

Note that in the example shown in FIG. 5, all of the status indicator bits have a value of one except for the status indicator bit transmitted by slave device 506c, which has a value of zero. Referring again to FIG. 4A, the master device 502 (which may, for example, be any of the devices 104a-d shown in FIG. 1) receives the status indicator message 504 over the bus 102 from the slave devices 506a-g (step 406). The master device 502 determines which of the slave devices 506a-g requires attention by examining the status indicator message 504 (step 406). In the example illustrated in FIG. 5, the master device 502 may determine that slave device 506c requires attention because the status indicator bit transmitted by slave device 506c (at bit position 5 in the

status indicator message 504) is the only bit in the status indicator message 504 that has a value of zero.

Particular embodiments of the processes illustrated in FIGS. 4A-4C will be described in more detail below. First, however, examples of techniques that may be used to simultaneously address multiple devices on the bus 102 will be described. In particular, examples of techniques will be described for transmitting a single message over the bus 102 that is received and processed by multiple devices on the bus 102.

For example, referring to FIG. 1, devices 104a-d include data bus interfaces 106a-d. As described above, these interfaces 106a-d may be standard interfaces designed for use with the bus 102. Devices 104a-d further include controllers 108a-d, respectively, which have been specially designed to transmit messages over the bus 102 to multiple devices simultaneously and/or to receive messages over the bus 102 that have been transmitted to multiple devices simultaneously.

To provide this ability, each of the devices 104a-d is provided with both a primary address and a secondary address. A particular primary address may be associated with one or more devices on the bus 102. For example, in one embodiment devices 104a-c share a common primary address 110 and have unique secondary addresses 112a-c (i.e., no two of the secondary addresses 112a-c are the same). As a result, primary address 110 identifies the group of devices 104a-c (referred to herein as a "device group"), while a combination of primary address 110 and one of the secondary addresses 112a-c uniquely identifies one of the devices 104a-c. For example, the combination of primary address 110 and secondary address 112b uniquely identifies device 104b.

Similarly, device 104d includes a primary address 116 that differs from primary address 110 and a secondary address

118. The combination of primary address 116 and secondary address 118, therefore, uniquely identifies device 104d.

As described in more detail below, however, it is not a requirement of the present invention that a primary-secondary address pair uniquely identify a single device. More generally, a primary address specifies a device group (a subset of the devices on the bus 102), and a secondary address specifies a subset of the device group specified by the primary address. As a result, a primary-secondary address pair may specify any number of devices.

In one embodiment, primary address 110 and primary address 116 are addresses implemented according to the protocol associated with the bus 102. For example, primary addresses 110 and 116 are shown within data bus interfaces 106a-d in FIG. 1 to indicate that such addresses are implemented in the same manner as addresses 124a-b in conventional devices 104e-f. As a result, messages transmitted over the bus 102 using primary address 110, for example, will be delivered to, received, and processed by all of devices 104a-c using the standard architecture and protocol of bus 102 (such as the I^2C bus architecture and protocol).

It should be appreciated that the particular number and combinations of primary and secondary addresses shown in FIG. 1 are provided merely for purposes of example and do not constitute limitations of the present invention. There may, for example, be any number of primary addresses and secondary addresses, and any number of devices may have a particular primary address. In one embodiment, however, primary addresses 110 and 116 are unique on the bus 102. In other words, each of the primary addresses 110 and 116 differs from each other and from the addresses 124a and 124b of the conventional devices 104e-f. The reason for making primary addresses 110 and 116 unique on the bus 102 will be described below.

Referring to FIGS. 2A-2B, techniques will now be described for addressing multiple devices simultaneously over the bus 102. Referring to FIG. 2A, a flowchart is shown of a method 200 that may be used by one of the devices 104a-d (referred to herein as the "master device") to transmit a message to one or more of the devices 104a-d over the bus 102. Referring to FIG. 2B, a flowchart is shown of a method 250 that may be used by one or more of the devices 104a-d (referred to herein as "slave devices") to receive a message that has been transmitted to one or more of the devices 104a-d over the bus 102. It should be appreciated that any one of the devices 104a-d may become a master device at a particular time by initiating the transmission of a message over the bus 102, in which case the remaining devices are considered slave devices for purposes of receipt of the message.

Referring to FIG. 2A, the master device addresses a first subset of the devices on the bus 102 using a primary address (step 202). For example, as described above, the first datum in a message transmitted over a data bus is typically an address that addresses a single device on the bus. The primary address transmitted over the bus 102 in step 202 may be transmitted as a conventional address according to the protocol associated with the bus 102.

Referring to FIG. 2B, each slave device receives the transmitted primary address over the bus 102 (step 252) and determines whether the transmitted primary address is the primary address of the slave device (step 254). If a slave device determines that the transmitted primary address is not the slave device's primary address, the slave device ignores the remainder of the message being transmitted by the master device (step 254). Steps 252-254 may be performed by the data bus interfaces 106a-d of the devices 104a-d.

Returning to FIG. 2A, the master device addresses a second subset of the devices on the bus 102 using a secondary

address (step 204). The second subset is a subset of the first subset addressed in step 252, and may include any number of devices (such as a single device in the first subset or all of the devices in the first subset). The master device may transmit the secondary address as, for example, a datum according to the protocol associated with the bus 102. For example, in the I^2C protocol, the datum following an address is treated as data to be delivered to the addressed device. The secondary address transmitted in step 204 may be transmitted as such a datum according to the I^2C protocol.

Although the secondary address is considered to be a datum according to the bus protocol, the slave devices addressed by the primary address receive this datum and interpret it as a secondary address. For example, returning to FIG. 2B, each slave device receives the transmitted secondary address over the bus 102 (step 256) and determines whether the transmitted secondary address is the slave device's secondary address (step 258). If the transmitted secondary address is the slave device's secondary address, the slave device becomes receptive to the remainder of the message transmitted by the master device. If not, the slave device determines whether the secondary address has a special value of "ALL," which indicates that all of the devices in the first subset (specified by the primary address received in step 252) are being addressed. The "ALL" value may be formatted in any way, one example of which is described in more detail below. If the transmitted secondary address has the special value of "ALL," the slave device becomes receptive to the remainder of the message transmitted by the master device. If not, the slave device ignores the remainder of the message.

Returning to FIG. 2A, the master device transmits information to the second subset of devices over the bus 102 (step 206). The information may be any information that may be transmitted over the bus 102, such as a command, data, or a

combination thereof. Returning to FIG. 2B, each of the slave devices addressed by the primary and secondary addresses received in steps 252 and 256 receives and processes the information transmitted over the bus 102 (step 262). Steps 256-262 may be performed by the controllers 108a-d of devices 104a-d.

It should be appreciated that the techniques described above with respect to FIGS. 2A-2B may be used to enable multiple devices to be addressed simultaneously over the bus 102 and to enable a single message to be transmitted simultaneously to multiple devices over the bus 102. A particular example will now be described in which the device 104d is the master device and in which devices 104a-c and 104e-f are slave devices.

In this example, the (master) device 104d transmits the primary address 110 over the bus 102 as an address according to the I^2C protocol (step 202). This addresses the devices 104a-c, which share the common primary address 110. The data bus interfaces of the (slave) devices 104a-c receive the transmitted primary address and determine whether it matches their own primary address (steps 252-254). Because the transmitted primary address is the primary address 110 shared by the devices 104a-c, the data bus interfaces 106a-c determine that there is a match, and the devices 104a-c become receptive to the remainder of the message being transmitted by the (master) device 104d.

It should be appreciated that the data bus interfaces 106e-f of the conventional devices 104e-f also compare the transmitted primary address 110 to the addresses 124a-b, respectively, and determine that there is no match. The conventional devices 104e-f therefore ignore the remainder of the message being transmitted by the (master) device 104d. Use of the techniques described herein with respect to FIGS.

2A-2B therefore do not interfere with the normal operation of conventional devices coupled to the bus 102.

Assume for purposes of example that (master) device 104d next transmits secondary address 112b over the bus 102 (step 204). The controller of each of the (slave) devices 104a-c receives the transmitted secondary address (step 256) and compares it to the device's own secondary address (step 258). The (slave) devices 104a and 104c will determine that the transmitted secondary address is not their secondary address (step 258), and that the transmitted secondary address does not have the value "ALL" (step 260), and will therefore ignore the remainder of the message transmitted by (master) device 104d. In contrast, (slave) device 104b will determine that the transmitted secondary address is the secondary address 112b of the (slave) device 104b (step 258), and will therefore become receptive to the remainder of the message transmitted by (master) device 104d. Conventional devices 104e-f will receive and ignore the secondary address transmitted in step 204.

Assume for purposes of example that (master) device 104d then transmits information (such as a command and/or data) over the bus 102 (step 206). The information is received and processed by (slave) device 104b (step 262); the information is received but ignored by the remaining devices 104a, 104c, and 104e-f.

It should be appreciated that if the (master) device 104d had transmitted the value "ALL" as the secondary address in step 204 in the example above, then all of the (slave) devices 104a-c would have received and processed the information transmitted by the (master) device 104d over the bus 102 (step 262).

Although in the examples above, the master device may address either one device in a device group (such as the device group consisting of devices 104a-c) or all of the

devices in a device group, it should be appreciated that in other embodiments the master device may address any number of devices in a device group. For example, the secondary address transmitted by the master device in step 204 may be encoded with a pattern that specifies a subset of the devices in the device group addressed by the primary address transmitted in step 202. Alternatively, each of the devices 104a-d may further be provided with a tertiary address, in which case a primary-secondary address pair may be used to specify a plural subset of a device group, and a primary-secondary-tertiary address triplet may be used to specify a single one of the devices in a device group. The techniques just described are provided merely for purposes of example and do not constitute limitations of the present invention.

Details of one embodiment in which the data bus 102 is an I^2C bus will now be described. Referring to FIG. 3A, a timing diagram 300 is shown that illustrates the normal structure of a message transmitted over an I^2C bus. The diagram 300 includes a graph 302 representing the serial data line (SDA) of the I^2C bus and a graph 304 representing the serial clock line (SCL) of the I^2C bus. The generation of clock signals on SCL is the responsibility of master devices; each master device generates it own clock signals when transferring data on the bus.

The message begins with a start bit 306 (in which both the SDA and SCL lines are held high), which indicates the initiation of a message by a master device. According to the I^2C-bus specification, a start condition is indicated by a HIGH to LOW transition on the SDA line while the SCL line is HIGH.

As defined by the I^2C-bus specification, the first byte of information following the start bit 306 is a 7-bit address 310 of a possible device on the bus. Address 310 is followed by a R/W bit 312, which indicates whether the addressed device is

to be read or written. A R/W value of zero indicates a write, while a R/W value of one indicates a read.

Each byte transmitted over an I^2C bus is followed by an acknowledge bit. For example, the byte consisting of address 310 and R/W bit 312 are following by an acknowledge bit 314. The acknowledge-related clock pulse is generated by the master. The receiving device must pull down the SDA line during the acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse. The receiver generates an acknowledge bit after each byte that it receives.

Following the acknowledge bit 314 is a data byte 316 transmitted by the transmitting device to the receiving device. The data byte 316 is followed by an acknowledge bit 318. This is followed by another data byte 320 and acknowledge bit 322. Although messages may include any number of data bytes, the two data bytes 316 and 320 are shown in FIG. 3A for purposes of example. The message concludes with a stop bit 324, indicated by a LOW to HIGH transition on the SDA line while the SCL line is HIGH.

Referring to FIG. 3B, a timing diagram 350 is shown that illustrates the structure of a message transmitted over an I^2C bus according to one embodiment of the present invention. The diagram 350 includes a graph 352 representing the serial data line (SDA) of the I^2C bus and a graph 354 representing the serial clock line (SCL) of the I^2C bus.

Note that the structure of the timing diagram 350 (FIG. 3B) is identical to the structure of the timing diagram 300 (FIG. 3A). What differs is the way in which the information transmitted over the bus 102 is interpreted and processed by the devices 104a-d.

More specifically, the message represented by timing diagram 350 begins with a start bit 356 that is identical to the start bit 306 shown in FIG. 3A. What follows is a primary address 360 (see step 202 in FIG. 2A), which has the same

structure as the conventional address 310 shown in FIG. 3A. As a result, the primary address 360 may be interpreted by conventional I^2C interfaces. Following this are a R/W bit 362 and an acknowledge bit 364 which are the same as the R/W bit 312 and the acknowledge bit 314, respectively. All devices having the primary address 360 will transmit acknowledge bit 314 over the bus 102.

Next is a secondary address 366 (see step 204 in FIG. 2A) having the same structure as the conventional data 316. As a result, the secondary address 366 may be interpreted by the controllers 108a-d of devices 104a-d but ignored by the data bus interfaces 106e-f of conventional devices 104e-f. Next is an acknowledge bit 368 that is identical to the acknowledge bit 318.

What follows is information 370 (see step 206 in FIG. 2A), which has the same structure as data 320. Although only one byte of information 370 is shown, there may be any number of bytes of information. Furthermore, information 370 may represent a command, data, or a combination thereof.

Following information 370 is an acknowledge bit 372 that is identical to acknowledge bit 322. Finally, there is a stop bit 374 that is identical to the stop bit 324.

It should be appreciated that using the techniques just described, a message that addresses multiple devices using a primary-secondary address pair may be transmitted over an I^2C bus using the standard I^2C bus architecture and protocol without interfering with the normal operation of other devices on the bus.

Some additional implementation issues related to the use of the I^2C bus architecture will now be described. Since the primary address 360 is a 7-bit address, the primary addresses 110 and 116 (FIG. 1) may be 7-bit addresses if the data bus 102 is an I^2C bus. Any 7-bit pattern may be used as a primary

address, so long as the pattern is not the I^2C address of any other device on the bus 102.

In one embodiment in which the data bus 102 is an I^2C bus, the secondary address 366 (FIG. 3B) actually contains both a command and a secondary address. More specifically, bits 1-4 of the secondary address 366 specify a command, while bits 5-7 specify a secondary address. In this embodiment, receipt by a slave device of a secondary address over the bus (step 256 in FIG. 2B) corresponds to receipt of bits 5-7 of secondary address 366. The slave device determines whether the received secondary address is the slave device's secondary address (step 258) by comparing bits 5-7 of secondary address 366 to the slave device's secondary address. If there is a match, the slave device decodes bits 1-4 of the secondary address 366 and executes the specified command (step 262). Executing the specified command may involve receiving additional data from the master device, such as data 320 (FIG. 3).

Bits 5-7 of the secondary address 366 may be used to encode up to eight different secondary addresses. In one embodiment, however, one value (such as 000 or 111) of bits 5-7 of the secondary is reserved for specifying the special "ALL" value described above with respect to FIG. 2B. As a result, bits 5-7 of a particular secondary address may specify either the value "ALL" or one or seven distinct secondary addresses. Various other techniques for encoding primary and secondary addresses will be apparent to those of ordinary skill in the art and are also within the scope of the present invention.

Bits 1-4 of the secondary address 366 may be used to specify any set of commands. It should be apparent that four bits may be used to specify up to sixteen distinct commands. Controllers 108a-d may be configured with appropriate hardware and/or software for transmitting, receiving, and/or executing commands specified by bits 1-4 of the secondary address 366.

If commands are encoded in the secondary address 366, as just described, a command will only be processed by a slave device if the primary address 360 is the primary address of the slave device and the R/W bit 362 is zero (write). Therefore, to read data from a slave device, in one embodiment the commands described above include an "output data" command. When a slave device reads this command, it will output one or more data bytes the next time its primary address is received followed by a R/W bit that is equal to one (read). The output data command should address a single slave device to avoid data collision and/or garbled data.

A particular embodiment for ascertaining the status of multiple devices on the bus 102 simultaneously, in which the bus 102 is an I^2C bus, will now be described in more detail. In this embodiment, each primary address is implemented as a 7-bit I^2C address. Assume, for example, that the primary address of all of the slave devices 506a-g shown in FIG. 5 is binary 1010000.

Furthermore, in this embodiment, each secondary address actually contains both a command and a secondary address, as described above. More specifically, bits 1-4 of each secondary address specifies a command, while bits 5-7 specify a secondary address. Secondary addresses are only received and processed by slave devices when the preceding primary address is followed by a read/write bit of zero.

The master device 502 may transmit the status request message to the slave devices 506a-g (FIG. 4A, step 402) by transmitting a special "Output Status" command to the slave devices 506a-g. This command may, for example, be implemented in bits 1-4 of the secondary addresses described above. Assume, for example, that the "Output Status" command is encoded as binary 1001. Assume further that the 3-bit secondary address 000 is used to implement the special value of "ALL" and thereby to address all of the slave devices 506a-

g (see FIG. 2B, step 260). To transmit the "Output Status" command to all of the slave devices 506a-g, the master device 502 may address the slave devices 506a-g using the primary address 1010000 followed by the secondary address 1001000 (where the first four bits "1001" specify the "Output Status" command and the last three bits "000" specify the special secondary address "ALL").

The "Output Status" command instructs each of the slave devices 506a-g to output a status indicator bit (as described above with respect to FIG. 4C) the next time the slave device receives its primary address (101000 in the present example) followed by a R/W bit having a value of 1 (read). Therefore, after transmitting the "Output Status" command to the slave devices 506a-g, the master device 502 may transmit over the bus 102 the primary address 1010000 followed by a R/W bit having a value of 1.

Upon receiving this combination of primary address and R/W bit, each of the slave devices 506a-g transmits its IRQ (Interrupt ReQuest for service) status bit over the bus 102 to form the status indicator message 504, as described above with respect to FIGS. 4B-4C and FIG. 5. In other words, in this embodiment a slave device uses its IRQ status bit to indicate its status. Therefore, to request attention, a slave device may set its IRQ status bit to zero and transmit the IRQ status bit in response to the "Output Status" command. The master device 502 may determine which of the slave devices 506a-g requires attention by examining the status indicator message 504 and determining which, if any, of the slave devices 506a-g transmitted an IRQ status bit having a value of zero (FIG. 4A, step 406).

Among the advantages of various embodiments of the present invention are one or more of the following.

The ability to ascertain the status of multiple devices on a bus simultaneously (i.e., using a single status indicator

message) may advantageously increase the speed with which the status of multiple devices on the bus may be ascertained. In particular, the ability to ascertain the status of multiple devices on a bus simultaneously may increase the speed with which a master device may determine which of multiple slave devices on the bus needs attention, thereby allowing the master device to provide service to the slave device needing attention more quickly.

Another advantage of various embodiments of the present invention is that the techniques employed are compatible with the architectures and protocols of conventional data buses, such as the I^2C bus. For example, as described above, the data bus interfaces 106a-d employed by the devices 104a-d may be conventional I^2C serial bus interfaces. The ability to use conventional data bus controllers in devices designed according to embodiments of the present invention may simplify the design and manufacture of such devices and reduce the cost of such design and manufacture.

Furthermore, the ability to work in conjunction with standard data buses is advantageous because devices constructed according to embodiments of the present invention may be deployed for use with standard data buses without requiring the modification of such buses or the conventional devices that are attached to them. As described above in particular with respect to FIGS. 2A-2B, the techniques employed by embodiments of the present invention do not interfere with the normal operation of devices on the bus 102. As a result, both conventional devices and devices constructed according to embodiments of the present invention may operate on a data bus at the same time without difficulty.

The devices 104a-d shown in FIG. 1 and described above may be any kind(s) of computing devices. Such devices include but are not limited to storage devices (such as hard disk drives and optical drives), output devices (such as monitors

and printers), power supplies, and input devices (such as keyboards and mice).

Although certain devices are referred to herein as "master devices" and others as "slave devices," it should be appreciated that the present invention is not limited to use in conjunction with architectures in which certain devices are designated as "master devices" and others as "slave devices." Rather, the term "master device" is used herein to refer more generally to a device that initiates communication with one or more other devices, and the term "slave device" is used herein to refer more generally to a device to which the initiation of communication is directed.

As used herein, the term "status" refers to any information indicative of the state of one or more devices coupled to the bus 102. For example, as described above, in one embodiment the "status" of a device refers to the state of the device's IRQ status bit, which in turn indicates whether the device is requesting interrupt service. The "status" of a device may, however, refer to any other aspect of the device's state.

Although the description herein may refer to ascertaining the status of multiple devices on the bus 102 "simultaneously," it should be appreciated that the present invention is not limited to embodiments in which the status of multiple devices are ascertained simultaneously. For example, in any bus architecture, a certain amount of time is required to transmit addresses and other data over the bus. The amount of time required depends on the length of the bus and other characteristics of the bus. As a result, particular information (such as the status indicator message 504) transmitted from one or more devices may reach the destination device at different times, depending on the transmission characteristics of the bus and the location of the devices on the bus. Furthermore, it should be appreciated that in

various embodiments of the present invention, the status of multiple devices is ascertained "simultaneously" in the sense that the status of such devices is indicated using a single message (such as the status indicator message 504) transmitted over the bus, rather than by sequentially transmitting messages each of which indicates the status of a particular device. The single message that indicates the status of multiple devices may, in particular implementations, be delivered to the master device over a period of time according to the particular implementations of the bus, the devices on the bus, and the particular requirements of the application.

Although in certain examples provided above the status request message (transmitted by the master device 502 in step 402 of FIG. 4A) may be described as a single message, the status request message may be a plurality of messages. The master device 502 may, for example, transmit a distinct status request message (such as the "Output Status" message described above) to each of the slave devices 506a-g. In response, the slave devices 506a-g may collectively transmit a single status indicator message, as described above with respect to step 454 of FIG. 4B. The ability to simultaneously address a plurality of devices on the bus 102 is therefore not a requirement of the present invention.

It should be appreciated that although each of the devices 104a-d is shown in FIG. 1 as having a distinct data bus interface and controller, the functionality of the data bus interface and controller in each of the devices may be implemented using a single interface or controller. For example, in one embodiment the controller and data bus interface of each of the devices 104a-d is implemented using the PIC16C72A microprocessor described above. More generally, the functionality provided by the controllers 108a-d and data bus interfaces 106a-d may be implemented using any appropriate hardware, software, or combination thereof.

The description above refers to the transmission of "messages" among devices 104a-f on the bus 102. As used herein, the term "message" refers to any unit of communication that may be transmitted by any device on the bus 102 to any one or more of the other devices on the bus 102. The protocol associated with the bus 102 typically defines a message format that is to be used to transmit messages on the bus. For example, a message typically includes an address followed by a command, which is sometimes followed by some amount of data. The command and/or data that follows the address is referred to above (e.g., in step 206 of FIG. 2A and step 262 of FIG. 2B) generically as "information" contained in the message. The present invention is not limited to transmission of any particular kind of message, or to transmission of messages using any particular bus architecture or protocol.

It is to be understood that although the invention has been described above in terms of particular embodiments, the foregoing embodiments are provided as illustrative only, and do not limit or define the scope of the invention. Various other embodiments, including but not limited to the following, are also within the scope of the claims.

What is claimed is: